CLAIMS

We claim:

1. A method for adding tamper resistance to a software program, the method comprising the step of:

installing a plurality of guards in a software program, each of said plurality of guards comprising at least one program instruction, wherein each of said plurality of guards is operable to verify the integrity of at least one program instruction of at least one other of said plurality of guards, and wherein the integrity of at least one program instruction of each of said plurality of guards is verified by at least one other of said plurality of guards.

2. The method of claim 1, further comprising the step of:

generating an executable version of said software program having said plurality of guards installed therein, so that said program instructions of said plurality of guards are executed by running said executable version of said software program.

3. The method of claim 1, wherein the step of installing a plurality of guards in a software program comprises, for each of said plurality of guards, the steps of:

storing a first integer a in said software program;

storing a multiplicative product pq of a second integer p and a third integer q in said software program;

identifying a code block in said software program, said code block comprising at least one program instruction of at least one other of said plurality of guards;

computing a value C for said code block, said value C being computed in a manner that makes it likely that said value C would change if said code block is modified;

computing a multiplicative inverse C', said multiplicative inverse C' satisfying the following:

$$C' * C \text{ modulo } (p - 1)(q - 1) = 1;$$

computing a constant R according to the following:

$$R = a^{C'} \text{ modulo } pq; \text{ and}$$

storing said constant R in said software program.

4. The method of claim 3, further comprising the steps of:

running said software program, thereby causing said program instructions of said plurality of guards to be executed; and

for at least one said code block in said software program,

while said software program is running, computing a value X for said at least one said code block, said value X being computed in the same manner as was said value C for said at least one code block, and

taking a defensive action if:

$$R^X \text{ modulo } pq \neq a.$$

5. The method of claim 1, wherein the step of installing a plurality of guards in a software program comprises, for each of said plurality of guards, the steps of:

selecting an asymmetric encryption key pair comprising a public key and a private key;

storing said public key in said software program;

identifying a code block in said software program, said code block comprising at least one program instruction of at least one other of said plurality of guards;

calculating a baseline value for said code block, said baseline value being computed in a manner that makes it likely that said value would change if said code block is modified;

encrypting said baseline value using said private key; and

storing said encrypted baseline value in said software program.

6. The method of claim 5, further comprising the steps of:

running said software program, thereby causing said program instructions of said plurality of guards to be executed; and

for at least one said code block in said software program,

while said software program is running, computing a runtime value of said at least one said code block, said runtime value being computed in the same manner as was said baseline for said at least said one code block,

decrypting said baseline value using said public key, and

taking a defensive action if said decrypted baseline value is not the same as said runtime value.

7. A method for producing tamper resistant copies of a software program, the method comprising the steps of:

installing a first watermark in a first copy of a software program;

installing a second watermark in a second copy of a software program;

installing a watermark guard in said first copy of a software program, said watermark guard comprising at least one program instruction, said watermark guard being operable to verify the integrity of said first watermark; and

installing said watermark guard in said second copy of said software program, said watermark guard being operable to verify the integrity of said second watermark, wherein said watermark guard is installed in the same location in said second copy of said software program as in said first copy of said software program.

8. The method of claim 7, further comprising

installing at least one other guard in said first copy of said software program and said second copy of said software program, each of said at least one other guards comprising at least one program instruction, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said watermark guard, wherein each said at least one other guard is the same in, and is installed in the same location in, said first copy of said software program and said second copy of said software program.

9. A method for producing a plurality of tamper resistant copies of a software program, the method comprising the steps of:

storing a first integer V in all of said plurality of copies of said software program;

storing a multiplicative product pq of a second integer p and a third integer q in all of said plurality of copies of said software program;

storing a one-way function $H()$ in all of said plurality of copies of said software program; and

for each of said plurality of copies of said software program,

storing a watermark W in said copy of said software program,

generating $H(W)$ comprising a result of executing one-way function $H()$ with watermark W comprising an input argument thereto,

computing a multiplicative inverse W' satisfying the following:

$$W' * H(W) \text{ modulo } (p - 1)(q - 1) = 1,$$

computing a constant Q according to the following:

$$Q = V^{W'} \text{ modulo } pq, \text{ and}$$

storing said constant Q in said copy of said software program.

10. The method of claim 9, further comprising, for at least one of said plurality of copies of said software program, the steps of:

running said at least one of said plurality of copies of said software program; and

taking a defensive action if:

$$Q^{H(W)} \text{ modulo } pq \neq V.$$

11. A method for producing a plurality of tamper resistant copies of a software programs, the method comprising the steps of:

selecting an asymmetric encryption key pair comprising a public key and a private key;

storing said public key in said software program in all of said plurality of copies of said software program;

storing a one-way function in all of said plurality of copies of said software program; and

for each of said plurality of copies of said software program,

storing a watermark in said copy of said software program,

generating a baseline value comprising the result of executing said one-way function with said watermark comprising an input argument thereto,

encrypting said baseline value with said private key, and

storing said encrypted baseline value in said software program.

12. The method of claim 11, further comprising, for at least one of said plurality of copies of said software program, the steps of:

running said at least one of said plurality of copies of said software program; and

while said executable version is running, generating a runtime value comprising the result of executing said one-way function with said watermark comprising an input argument thereto;

decrypting said baseline value using said public key; and

taking a defensive action if said runtime value differs from said decrypted baseline value.

13. A method for producing a tamper resistant software program comprising mutually reliant program parameters, the method comprising the steps of:

installing two or more program parameters in a software program;

generating a value comprising said two or more program parameters, said value being likely to change if one or more of said program parameters is changed;

storing said value in said software program; and

installing a program parameter guard in said software program, said program parameter guard comprising at least one program instruction, said program parameter guard being operable to verify the integrity of said value.

14. The method of claim 13, further comprising

installing in said software program at least one other guard, each of said at least one other guards comprising at least one program instruction, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said program parameter guard.

15. The method of claim 13, wherein said value is generated by concatenating said program parameters.

16. A method for producing a software program comprising mutually reliant program parameters, the method comprising the steps of:

storing a first integer V in a software program;

storing a multiplicative product pq of a second integer p and a third integer q in said software program;

storing a one-way function $H()$ in said software program;

storing one or more program parameters in said software program;

generating baseline constant U comprising two or more of said program parameters;

generating baseline value J, said baseline value J comprising a result of executing one-way function $H()$ with baseline constant U comprising an input argument thereto;

computing a multiplicative inverse U' satisfying the following:

$$U' * J \text{ modulo } (p - 1)(q - 1) = 1;$$

computing a constant Q according to the following:

$$Q = V^{U'} \text{ modulo } pq; \text{ and}$$

storing said constant Q in said software program.

17. The method of claim 16, further comprising the steps of:

running said software program;

while said software program is running, generating runtime constant U comprising two or more of said program parameters;

generating runtime value K, said runtime value K comprising a result of executing one-way function $H()$ with runtime constant U comprising an input argument thereto; and

taking a defensive action if:

$$Q^K \text{ modulo } pq \neq V.$$

18. A method for producing a software program comprising mutually reliant program parameters, the method comprising the steps of:

selecting an asymmetric encryption key pair comprising a public key and a private key;

storing one or more program parameters in said software program;

storing said public key in said software program;

generating baseline constant *U* comprising two or more of said program parameters;

generating baseline value *J*, said baseline value *J* comprising a result of executing one-way function *H()* with baseline constant *U* comprising an input argument thereto;

encrypting said baseline value *J* with said private key; and

storing said encrypted baseline value *J* in said software program.

19. The method of claim 18, further comprising the steps of:

running said software program;

while said software program is running, generating runtime constant *U* comprising two or more of said program parameters;

generating runtime value *K*, said runtime value *K* comprising a result of executing one-way function *H()* with runtime constant *U* comprising an input argument thereto;

decrypting said baseline value *J* using said public key; and

taking a defensive action if said runtime value *K* differs from said decrypted baseline value *J*.

20. A method for producing a plurality of tamper resistant copies of a software program wherein each copy of said software program comprises mutually reliant program parameters, the method comprising the steps of:

installing two or more program parameters in a first copy of a software program;

generating a first value comprising said two or more program parameters installed in said first copy of said software program, said first value being likely to change if one or more of said program parameters installed in said first copy of said software program is changed;

storing said first value in said first copy of said software program;

installing two or more program parameters in a second copy of a software program;

generating a second value comprising said two or more program parameters installed in said second copy of said software program, said second value being likely to change if one or more of said program parameters installed in said second copy of said software program is changed;

storing said second value in said second copy of said software program;

installing a program parameter guard in said first copy of a software program, said program parameter guard comprising at least one program instruction, said program parameter guard being operable to verify the integrity of said first value; and

installing said program parameter guard in said second copy of said software program, said program parameter guard being operable to verify the integrity of said second value, wherein said program parameter guard is installed in the same location in said second copy of said software program as in said first copy of said software program.

21. The method of claim 20, further comprising

installing, in said first copy of said software program and said second copy of said software program, at least one other guard, each of said at least one other guards comprising at least one program instruction, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said program parameter guard, wherein each said at least one other guard is the same in, and is installed in the same location in, said first copy of said software program and said second copy of said software program.

22. A method for adding tamper resistance to a software program, the method comprising the steps of:

identifying a first code block in a software program;

creating a second code block, said second code block comprising a copy of said first code block;

disguising said second code block; and

installing at least one repair guard in said software program, each of said at least one repair guards comprising at least one program instruction.

23. The method of claim 22, further comprising the steps of:

executing said at least one program instruction of at least one of said at least one repair guards;

undisguising said second code block; and

overwriting said first client code block with said undisguised second code block copy.

24. The method of claim 22, further comprising, after the step of creating said second code block, the step of:

damaging said first code block so that said first code bock causes said software program to execute improperly.

25. The method of claim 24, further comprising the steps of:

executing said at least one program instruction of at least one of said at least one repair guards;

undisguising said second code block; and

overwriting said first client code block with said undisguised second code block copy.

26. A method for adding tamper resistance to a software program, the method comprising the steps of:

installing a plurality of guards in a software program, each of said plurality of guards comprising at least one guard program instruction; and

installing guard selection program instructions in said software program, said guard selection program instructions being operable to alter the control flow of said software program.

27. The method of claim 26, wherein said guard selection program instructions, when executed during running of said software program, alter the control flow of said software program by causing the execution of one or more of said guard program instructions to be skipped.

28. The method of claim 26, wherein one or more of said guard program instructions ordinarily would not be executed when running said software program, and where said guard selection program instructions, when executed during running of said software program, alter the control flow of said software program by causing at least one of said one or more of said guard program instructions to be executed.

29. A method for adding tamper resistance to a software program, the method comprising the steps of:

installing two or more repair guards in a software program, each said repair guard comprising one or more program instruction that, when executed during running of said software program, are operable to overwrite one or more program instructions of said software program with properly functioning program instructions;

installing guard selection program instructions in said software program, said guard selection program instructions, when executed during said running of said software program, causing one or more of said repair guards to be executed.

30. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a plurality of guards installed in a software program, each of said plurality of guards comprising at least one program instruction, wherein each of said plurality of guards is operable to verify the integrity of at least one program instruction of at least one other of said plurality of guards, and wherein the integrity of at least one program instruction of each of said plurality of guards is verified by at least one other of said plurality of guards.

31. A plurality of copies of a recordable computer readable media having a tamper resistant software program thereon, said plurality of copies comprising:

a first watermark installed in a first copy of a software program recorded on a first copy of a recordable computer readable media;

a second watermark installed in a second copy of said software program recorded on a second copy of said recordable computer readable media;

a watermark guard installed in said first copy of said software program, said watermark guard comprising at least one program instruction, said watermark guard being operable to verify the integrity of said first watermark; and

said watermark guard installed in said second copy of said software program, said watermark guard being operable to verify the integrity of said second watermark;

and wherein said watermark guard is the same in, and is installed in the same location in, said first copy of said software program and said second copy of said software program.

32. The plurality of copies of the recordable computer readable media of claim 31, further comprising:

at least one other guard installed in said first copy of said software program and in said second copy of said software program, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said watermark guard, wherein each said at least one other guard is the same in, and is installed in the same location in, said first copy of said software program and said second copy of said software program.

33. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

two or more program parameters installed in a software program;

a value comprising said two or more program parameters stored in said software program;

a program parameter guard installed in said software program, said program parameter guard comprising at least one program instruction, said program parameter guard being operable to verify the integrity of said value.

34. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 33, further comprising

at least one other guard installed in said software program, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said program parameter guard.

35. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 33, wherein said value is generated by concatenating said program parameters.

36. A plurality of copies of a recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

two or more program parameters installed in a first copy of a software program recorded on a first copy of a recordable computer readable media;

a first value installed in said first copy of said software program, said first value comprising said two or more program parameters;

two or more program parameters installed in a second copy of said software program recorded on a second copy of said recordable computer readable media;

a second value installed in said second copy of said software program, said second value comprising said two or more program parameters;

a program parameter guard installed in said first copy of a software program, said program parameter guard comprising at least one program instruction, said program parameter guard being operable to verify the integrity of said first value; and

said program parameter guard installed in said second copy of said software program, said program parameter guard being operable to verify the integrity of said second value;

and wherein said program parameter guard is the same in, and is installed in the same location in, said first copy of said software program and said second copy of said software program.

37. The plurality of copies of a recordable computer readable media having a tamper resistant software program recorded thereon of claim 32, further comprising:

at least one other guard installed in said first copy of said software program and said second copy of said software program, wherein at least one of said at least one other guards is operable to verify the integrity of at least one program instruction of said program parameter guard, wherein at least one other guard is/are the same in, and is/are installed in the same location(s) in, said first copy of said software program and said second copy of said software program.

38. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a first code block in a software program;

a second code block in said software program, said second code block comprising a disguised copy of said first code block; and

at least one repair guard installed in said software program, at least one of said at least one repair guards comprising one or more program instructions operable when executed to automatically undisguised said second code block and overwrite said first code block with said undisguised second code block.

39. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 38, wherein said first code block is damaged so that said software program executes improperly.

40. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a plurality of guards installed in a software program, each of said plurality of guards comprising at least one guard program instruction; and

guard selection program instructions, said guard selection program instructions being operable to alter a control flow of said software program.

41. The recordable computer readable media having a software program recorded thereon of claim 40, wherein said guard selection program instructions are operable to alter said control flow of said software program by causing the execution of one or more of said guard program instructions to be skipped.

42. The recordable computer readable media having a software program recorded thereon of claim 40, wherein said guard selection program instructions are operable to alter said control flow of said software program by causing the execution of at least one of said one or more of said guard program instructions that otherwise would be skipped.

43. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

two or more repair guards installed in a software program, each said repair guard comprising one or more program instruction that, when executed during execution of said software program, are operable to overwrite one or more program instructions of said software program with properly functioning program instructions;

guard selection program instructions installed in said software program, said guard selection program instructions, when executed during said running of said software program, causing one or more of said repair guards to be executed.

44. A method for adding tamper resistance to a software program, said software program comprising a first software program variable and a second software program variable, wherein said first software program variable does not depend on said second software program variable, the method comprising the step of:

installing at least one guard in said software program, said at least one guard linking said first software program variable and said second software program variable so that an unauthorized change to said first software program variable changes said second software program variable.

45. A method for adding tamper resistance to a software program, said software program comprising a first code block and a second code block, wherein said first code block functions independently of said second code block, the method comprising the step of:

installing at least one guard in said software program, said at least one guard causing functioning of said second code block to depend on proper functioning of said first code block so that an unauthorized change to said first code block changes said functioning of said second code block.

46. A method for adding tamper resistance to a software program, the method comprising:

installing a silent guard in a software program, said silent guard comprising one or more program instructions.

47. The method of claim 46, further comprising the step of:

evaluating the integrity of one or more data items in computer memory when said software program is running; and

taking a defensive action if said silent guard detects a deficiency in said integrity of said one or more data items.

48. The method of claim 46, further comprising the step of:

assigning, by operation of said silent guard, a predetermined value to a software program variable in computer memory when said software program is running before said software program variable is used in a computation.

49. The method of claim 46, wherein said silent guard comprises a variable whose value is computed during execution of said software program, and wherein a defensive action results if said silent guard detects an unexpected value of said variable.

50. The method of claim 49, wherein said variable has an expected value, the method further comprising the step of:

comparing a runtime value of said variable against said expected value.

51. The method of claim 50, wherein the step of comparing a runtime value of said variable against said expected value comprises the step of:

inserting one or more mathematical expressions into one or more program instructions in said software program, said one or more mathematical expressions comprising said runtime value of said variable and said expected value of said variable, wherein correct execution of said one or more program instructions depends on said runtime value of said variable being the same as said expected value of said variable.

52. The method of claim 50, wherein the step of comparing a runtime value of said variable against said expected value comprises the step of:

comparing said expected value after said expected value is processed through an algorithm with said runtime value after said runtime value is processed through said algorithm.

53. The method of claim 46, wherein said value of said variable changes at least once during program execution, the method further comprising the step of:

determining a runtime value of said variable at a point in software program execution; and

comparing said runtime value of said variable at said point in software program execution against an expected value of said variable at said point in software program execution.

54. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a first software program variable;

a second software program variable; and

at least one guard, said at least one guard linking said first software program variable and said second software program variable, such that an unauthorized change to said first software program variable changes said second software program variable.

55. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a first code block;

a second code block; and

at least one guard, said at least one guard causing functioning of said second code block to depend on proper functioning of said first code block, such that an unauthorized change to said first code block changes said functioning of said second code block.

56. A recordable computer readable media having a tamper resistant software program recorded thereon, comprising:

a software program comprising one or more program instructions; and

a silent guard comprising one or more guard program instructions, said one or more guard program instructions installed in said software program.

57. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 56, wherein said one or more guard program instructions are operable to evaluate the integrity of one or more data items in computer memory when said software program is running, and to take a defensive action if a deficiency in said integrity of said one or more data items is detected.

58. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 56, comprising:

at least one software program variable, said one or more guard program instructions being operable to assign a predetermined value to at least one of said at least one software program variables in computer memory before said software program variable is used in a computation during execution of said software program.

59. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 56, wherein said software program comprises a variable whose value is computed during execution of said software program, and wherein a defensive action results if an unexpected value of said variable is detected.

60. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 59, wherein said variable has an expected value, and wherein said defensive action results if a runtime value of said variable is not the same as said expected value.

61. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 59, further comprising:

one or more mathematical expressions inserted into one or more program instructions in said software program, said one or more mathematical expressions comprising said runtime value of said variable and said expected value of said variable, wherein correct execution of said one or more program instructions depends on said runtime value of said variable being the same as said expected value of said variable.

62. The recordable computer readable media having a tamper resistant software program recorded thereon of claim 59, further comprising:

an algorithm, wherein said defensive action results if a runtime value of said variable after said runtime value is processed through said algorithm is not the same as

said expected value after said expected value is processed through said algorithm.

63. A recordable computer readable media having a computer program for adding tamper resistant features to a software program recorded thereon, comprising:

program instructions operable to install a plurality of guards in said software program, each of said plurality of guards comprising at least one guard program instruction, wherein each of said plurality of guards is operable to verify the integrity of at least one guard program instruction of at least one other of said plurality of guards, and wherein the integrity of at least one guard program instruction of each of said plurality of guards is verified by at least one other of said plurality of guards; and

program instructions operable to generate an executable version of said software program having said plurality of guards installed therein.

64. A recordable computer readable media having a computer program for adding tamper resistant features to a software program recorded thereon, comprising:

program instructions operable to install a first watermark in a first copy of a software program;

program instructions operable to install a second watermark in a second copy of a software program;

program instructions operable to install a watermark guard in said first copy of a software program, said watermark guard comprising at least one guard program instruction, said watermark guard being operable to verify the integrity of said first watermark; and

program instructions operable to install said watermark guard in said second copy of said software program, said watermark guard being operable to verify the integrity of said second watermark, wherein said watermark guard is installed in the same location in said second copy of said software program as in said first copy of said software program.

65. A recordable computer readable media having a computer program for adding tamper resistant features to a software program recorded thereon, wherein each copy of said software program comprises mutually reliant program parameters, comprising:

program instructions operable to install two or more program parameters in a first copy of a software program;

program instructions operable to generate a first value comprising said two or more program parameters installed in said first copy of said software program, said first value being likely to change if one or more of said program parameters installed in said first copy of said software program is changed;

program instructions operable to store said first value in said first copy of said software program;

program instructions operable to install two or more program parameters in a second copy of a software program;

program instructions operable to generate a second value comprising said two or more program parameters installed in said second copy of said software program, said second value being likely to change if one or more of said program parameters installed in said second copy of said software program is changed;

program instructions operable to store said second value in said second copy of said software program;

program instructions operable to install a program parameter guard in said first copy of a software program, said program parameter guard comprising at least one program instruction, said program parameter guard being operable to verify the integrity of said first value; and

program instructions operable to install said program parameter guard in said second copy of said software program, said program parameter guard being operable to verify the integrity of said second value, wherein said program parameter guard is installed in the same location in said second copy of said software program as in said first copy of said software program.

66. A recordable computer readable media having a computer program for adding tamper resistant features to a software program recorded thereon, comprising:

program instructions operable to identify a first code block in a software program;

program instructions operable to create a second code block, said second code block comprising a copy of said first code block;

program instructions operable to disguise said second code block; and

program instructions operable to install at least one repair guard in said software program, each of said at least one repair guards comprising at least one guard program instruction.